Name-Tejas Gadge Roll No-TYITA33

// code to see orphan and zombie process

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h> //
sorting in ascending
void bubbleSort_ascending(int array[], int size) {
for (int step = 0; step < size - 1; ++step) {
for (int i = 0; i < size - step - 1; ++i) {
if (array[i] > array[i + 1]) {         int temp =
array[i];         array[i] = array[i + 1];
array[i + 1] = temp;
      }
    }
  }
}
//descending sort
void bubbleSort_descending(int array[], int size) {
for (int step = 0; step < size - 1; ++step) {
for (int i = 0; i < size - step - 1; ++i) {
if (array[i] < array[i + 1]) {         int temp =
array[i];         array[i] = array[i + 1];
array[i + 1] = temp;
      }
    }
  }
}

// taking input
void print_Array(int array[], int size) {
for  (int  i  =  0;  i  <  size;  ++i)  {
printf("%d  ", array[i]);
  }
  printf("\n");
}
// to print element by parent process void
print_parent(int array[], int size)
{
       for(int i = 0; i <size; i++)
       {
  printf("%d (Process Id = %d with ppId = %d)\n", array[i], getpid(),
getppid());
               sleep(1);
       }

}
// to print element by child process void
print_child(int array[], int size) {
```

```
        for(int i = 0; i <size; i++)
        {
            printf("    %d (Process Id = %d with ppId = %d)\n",
array[i], getpid(), getppid());
                sleep(2);
        }
}
int main()
{
        int size;
        printf("Enter the number of elements : ");
        scanf("%d", &size);
        int a[size];
        printf("Enter array elements : \n");

        for(int i = 0; i < size; i++)
        {
                printf("a[%d] : ", i+1);
                scanf("%d", &a[i]);
        }
        int p = fork();
        if(p > 0)
        {
            bubbleSort_ascending(a, size);
// printArray(a, size);
print_parent(a, size);
                //wait(NULL);
                 //sleep(300);
                printf("parent terminating\n");
        }
        else if (p==0)
        {
            bubbleSort_descending(a, size);
// printArray(a, size);           print_child(a,
size);
               printf("child terminating\n");
        }
        else
        {
                printf("Child is not created");
        }
}
```

```
tejas@LAPTOP-Q2068G6F:~/first$ cd os_assignment/
tejas@LAPTOP-Q2068G6F:~/first/os_assignment$ gcc assignment2.c
tejas@LAPTOP-Q2068G6F:~/first/os_assignment$ ./a.out
Enter the number of elements : 6
Enter array elements :
a[1] : 89
a[2] : 56
a[3] : -45
a[4] : 10
a[5] : 94
a[6] : 10
-45 (Process Id = 266 with ppId = 10)
        94 (Process Id = 267 with ppId = 266)
        89 (Process Id = 267 with ppId = 266)
10 (Process Id = 266 with ppId = 10)
        56 (Process Id = 267 with ppId = 266)
10 (Process Id = 266 with ppId = 10)
        10 (Process Id = 267 with ppId = 266)
56 (Process Id = 266 with ppId = 10)
        10 (Process Id = 267 with ppId = 266)
89 (Process Id = 266 with ppId = 10)
94 (Process Id = 266 with ppId = 10)
        -45 (Process Id = 267 with ppId = 266)
parent terminating
child terminating
tejas@LAPTOP-Q2068G6F:~/first/os_assignment$
```

```
tejas@LAPTOP-Q2068G6F:~$ ps
  PID TTY          TIME CMD
  366 pts/1    00:00:00 bash
  387 pts/1    00:00:00 ps
tejas@LAPTOP-Q2068G6F:~$ ps -aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0   1756  1072 ?        Sl   22:36   0:00 /init
root         8  0.0  0.0   2112   344 ?        Ss   22:36   0:00 /init
root         9  0.0  0.0   2112   352 ?        S    22:36   0:00 /init
tejas       10  0.0  0.1  10172  5308 pts/0    Ss   22:36   0:00 -bash
root       364  0.0  0.0   2112   344 ?        Ss   22:44   0:00 /init
root       365  0.0  0.0   2112   352 ?        S    22:44   0:00 /init
tejas      366  0.0  0.1  10040  5072 pts/1    Ss   22:44   0:00 -bash
tejas      385  0.0  0.0   2492   508 pts/0    S+   22:45   0:00 ./a.out
tejas      386  0.0  0.0      0     0 pts/0    Z+   22:45   0:00 [a.out] <defunct>
tejas      388  0.0  0.0  10832  3344 pts/1    R+   22:45   0:00 ps -aux
tejas@LAPTOP-Q2068G6F:~$
```